Proof interpretations: a modern perspective

Lecture 1 - Hilbert's program and the rise of proof theory

Anupam Das & Thomas Powell

University of Copenhagen & Technische Universität Darmstadt

North American Summer School on Logic, Language, and Information

Carnegie Mellon University

25 June 2018

These slides are available at `http://www.anupamdas.com/nasslli18`.

Some references

Main references for this course:

- Avigad, J. and Feferman, S. (1998). Gödel's functional ("Dialectica") interpretation. In Buss, S. R., editor, *Handbook of Proof Theory*, volume 137, pages 337–405. Elsevier. http://www.andrew.cmu.edu/user/avigad/Papers/dialect.pdf
- Kohlenbach, U. (2008). *Applied Proof Theory - Proof Interpretations and their Use in Mathematics.* Springer Monographs in Mathematics. Springer
- Stanford Encyclopedia of Philosophy. http://plato.stanford.edu/.

(More detailed bibliography for each lecture can be found at end of slides.)

Structure of course

Historical introduction

Today (A+T): Hilbert's program and the rise of proof theory

In-depth exploration

Tuesday (A): Higher-order computation.

Wednesday (T): The functional interpretation of Intuitionistic Arithmetic.

Thursday (T): Classical logic and the negative translation.

Highlights

Friday (A+T): Proof interpretations today.

Outline

“Concrete” mathematics

Up to the 19^{th} century, mathematics was primarily concerned with concrete objects which could be explicitly constructed. E.g.:

- Every number is either even or odd.
- There are infinitely many prime numbers.
- Every number can be written as the sum of four squares.
- Every continuous function can be integrated.
- Every non-constant polynomial over over the complex numbers has a root.

"Concrete" mathematics

Up to the 19^{th} century, mathematics was primarily concerned with concrete objects which could be explicitly constructed. E.g.:

- Every number is either even or odd.
- There are infinitely many prime numbers.
- Every number can be written as the sum of four squares.
- Every continuous function can be integrated.
- Every non-constant polynomial over over the complex numbers has a root.

Proofs of these results yield concrete algorithms. E.g.:

- We can decide whether a number is even or odd.
- We can find the next prime number.
- For any number we can find four squares which sum to that number.
- We can compute the integral of f up to any desired accuracy.
- We can compute roots of polynomials up to any desired accuracy.

Infinitude of primes

Proposition

There are infinitely many prime numbers.

Fundamental theorem of arithmetic: every number has a prime factorisation.

Proof (Aristotle, Euclid).

Suppose there are only finitely many primes and label them $p_1, \ldots, p_k$. Apply the fundamental theorem to $p_1 \cdot \cdots \cdot p_k + 1$ to find a prime factor. This cannot be any of the p_i. □

Infinitude of primes

Proposition

There are infinitely many prime numbers.

Fundamental theorem of arithmetic: every number has a prime factorisation.

Proof (Aristotle, Euclid).

Suppose there are only finitely many primes and label them $p_1, \ldots, p_k$. Apply the fundamental theorem to $p_1 \cdot \cdots \cdot p_k + 1$ to find a prime factor. This cannot be any of the p_i. □

What constructive information can we extract from this proof?

- The fundamental theorem of arithmetic gives us a factoring algorithm.
- Given primes $p_1, \ldots, p_k$, we simply factor $p_1 \cdots p_k + 1$ to find a new prime.

In other words, the proof comes equipped with an algorithm for finding the next prime.

We also derive a bound: for any number n there is a prime p with $n < p \leq n! + 1$.

Non-constructive mathematics

With the advent of modern mathematics in the 19$^{\text{th}}$ century, mathematicians started to reason about non-constructible objects.

- For every $f : \mathbb{N} \to \mathbb{N}$ there exists an n such that $f(n) \leq f(m)$ for all m.
- Every set of real numbers has a least upper bound.
- Every monotone, bounded sequence converges to a limit.
- Every ring has a maximal ideal.
- Every vector space has a basis.

Non-constructive mathematics

With the advent of modern mathematics in the 19$^{\text{th}}$ century, mathematicians started to reason about non-constructible objects.

- For every $f : \mathbb{N} \to \mathbb{N}$ there exists an n such that $f(n) \leq f(m)$ for all m.
- Every set of real numbers has a least upper bound.
- Every monotone, bounded sequence converges to a limit.
- Every ring has a maximal ideal.
- Every vector space has a basis.

In general, none of these existence results yield effective algorithms.

They give the existence of *ideal objects*, based purely on formal reasoning.

Non-constructive mathematics

With the advent of modern mathematics in the 19$^{\text{th}}$ century, mathematicians started to reason about non-constructible objects.

- For every $f : \mathbb{N} \to \mathbb{N}$ there exists an n such that $f(n) \leq f(m)$ for all m.
- Every set of real numbers has a least upper bound.
- Every monotone, bounded sequence converges to a limit.
- Every ring has a maximal ideal.
- Every vector space has a basis.

In general, none of these existence results yield effective algorithms.

They give the existence of *ideal objects*, based purely on formal reasoning.

Question: Do these ideal objects really 'exist'?

Intuitively, we believe they do since we trust mathematical reasoning. But some rather bizarre phenomena may occur...

Example: irrational powers

Proposition

There are irrational numbers a, b such that a^b is rational.

Example: irrational powers

Proposition

There are irrational numbers a, b such that a^b is rational.

Proof.

We know that $\sqrt{2}$ is irrational. What about $\sqrt{2}^{\sqrt{2}}$? We have two cases:

- If $\sqrt{2}^{\sqrt{2}}$ is rational, then set $a = b = \sqrt{2}$.
- Otherwise, set $a = \sqrt{2}^{\sqrt{2}}$ and $b = \sqrt{2}$. We have,

$$a^b = (\sqrt{2}^{\sqrt{2}})^{\sqrt{2}} = \sqrt{2}^{\sqrt{2}\cdot\sqrt{2}} = \sqrt{2}^2 = 2$$

so a^b is rational as required. □

Example: irrational powers

Proposition

There are irrational numbers a, b such that a^b is rational.

Proof.

We know that $\sqrt{2}$ is irrational. What about $\sqrt{2}^{\sqrt{2}}$? We have two cases:

- If $\sqrt{2}^{\sqrt{2}}$ is rational, then set $a = b = \sqrt{2}$.
- Otherwise, set $a = \sqrt{2}^{\sqrt{2}}$ and $b = \sqrt{2}$. We have,

$$a^b = (\sqrt{2}^{\sqrt{2}})^{\sqrt{2}} = \sqrt{2}^{\sqrt{2}\cdot\sqrt{2}} = \sqrt{2}^2 = 2$$

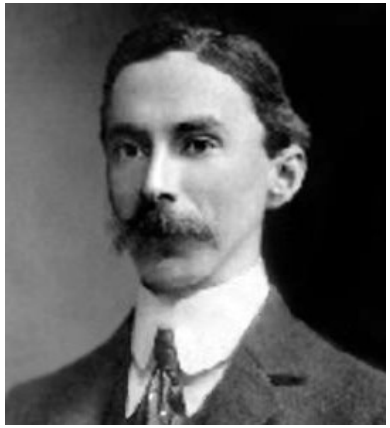
so a^b is rational as required. □

We have proved the proposition, but we do not know which of the cases hold!

Question: Is $\sqrt{2}^{\sqrt{2}}$ rational or irrational?!

Down the rabbit hole...

Even worse, this style of mathematical reasoning can lead us down fallacious paths.



Example (Russell's paradox (B. Russell, 1901))

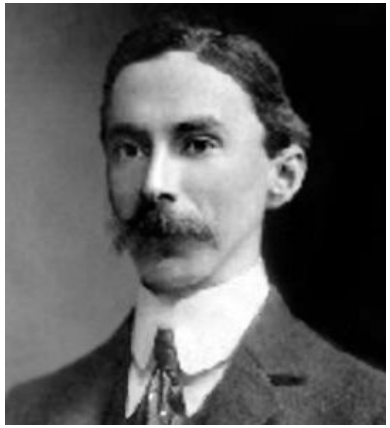
Let $R := \{x : x \notin x\}$. Is $R \in R$?

- If $R \in R$ then by definition we must have that $R \notin R$;
- But if $R \notin R$ then we must have $R \in R$.

We have a contradiction!

Down the rabbit hole...

Even worse, this style of mathematical reasoning can lead us down fallacious paths.



Example (Russell's paradox (B. Russell, 1901))

Let $R := \{x : x \notin x\}$. Is $R \in R$?

- If $R \in R$ then by definition we must have that $R \notin R$;
- But if $R \notin R$ then we must have $R \in R$.

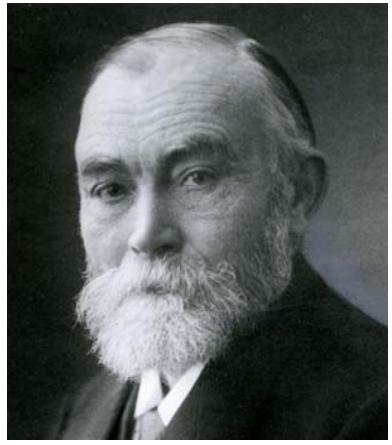
We have a contradiction!

Conclusion: Our naive formulation of mathematics, in particular set theory, is *inconsistent*, and so cannot be trusted.

This one simple paradox destroyed a life's work:

Corollary

Gottlob Frege's foundations of arithmetic are inconsistent.



The foundational crisis

The tragedy of Frege's foundations, destroyed by Russell's paradox led to the so-called foundational crisis at the turn of the century:

- Can we construct solid formal foundations for mathematics?
- Can we ensure that they are consistent, and do no succumb to paradoxes?
- In particular, can we give a formal treatment of set theory?

The foundational crisis

The tragedy of Frege's foundations, destroyed by Russell's paradox led to the so-called foundational crisis at the turn of the century:

- Can we construct solid formal foundations for mathematics?
- Can we ensure that they are consistent, and do no succumb to paradoxes?
- In particular, can we give a formal treatment of set theory?

These led to the resurgence of fundamental philosophical questions:

- What *is* a mathematical proof?
- Can mathematics/arithmetic be reduced to pure logic?
- Do mathematical objects 'exist' in some abstract sense, or are they just symbols on a piece of paper?

The foundational crisis

The tragedy of Frege's foundations, destroyed by Russell's paradox led to the so-called foundational crisis at the turn of the century:

- Can we construct solid formal foundations for mathematics?
- Can we ensure that they are consistent, and do no succumb to paradoxes?
- In particular, can we give a formal treatment of set theory?

These led to the resurgence of fundamental philosophical questions:

- What *is* a mathematical proof?
- Can mathematics/arithmetic be reduced to pure logic?
- Do mathematical objects 'exist' in some abstract sense, or are they just symbols on a piece of paper?

A historical note: Set-theoretic paradoxes were arguably known already by the 1880s, e.g. the *Burali-Forti* paradox. However, the sheer simplicity of Russell's paradox shook the mathematical world, questioning the most basic principles of mathematical reasoning.

Outline

Emerging philosophies

Two competing schools of thought emerged as a reaction to the foundational crisis:

INTUITIONISM (led by L. E. J. Brouwer, see [Iemhoff, 2016])

- Based on *semantics*.
- Mathematics is a mental construction: Something exists only if it can be exhibited.
- Infinite sets, maximals ideals, limits are all dubious unless they can be built explicitly.

FORMALISM (led by D. Hilbert, see [Weir, 2015])

- Based on *syntax*.
- Mathematics is a game of symbols: something 'exists' if it can be derived from mathematical axioms by logical inference rules.
- Infinite sets, maximal ideals, limits are all fine, as long as our underlying logical system *can be trusted*.

We will follow the formalist approach, but take advantage of elegant ideas from both.

Hilbert's Program

In the early 20th century, Hilbert proposed a set of benchmarks for a satisfactory foundation of arithmetic:

Hilbert's Program, 1921

Find a collection of axioms and inference rules $\mathcal{P}$ for arithmetic which is:

1. ***Complete:*** *all true statements in the language of arithmetic are provable in $\mathcal{P}$.*
2. ***Consistency:*** *$\mathcal{P}$ does not prove a contradiction.*

Aside: But this statement is circular! To show that $\mathcal{P}$ is consistent we need to work in some other system, which in turn needs to be shown to be consistent etc. Hilbert was well aware of this, so he asked for the following further refinement:

(continued)

3. ***Finitary consistency:*** *the fact that $\mathcal{P}$ is consistent is demonstrable using only simple finitary methods, whose validity cannot be questioned.*

Unwinding Hilbert's formalism

The notion of 'finitary' in Hilbert's program is crucial and, indeed, a subject of debate. Hilbert asks us to distinguish object-level systems $\mathcal{P}$ from 'finitary' meta-level systems $\mathcal{N}$ where:

- $\mathcal{P}$ can reason about crazy objects which cannot be constructed (and which would be rejected by intuitionists).
- $\mathcal{N}$ is grounded in a world of numbers and simple arithmetic operations, which no reasonable person could doubt. It is unquestionably correct.

Unwinding Hilbert's formalism

The notion of 'finitary' in Hilbert's program is crucial and, indeed, a subject of debate. Hilbert asks us to distinguish object-level systems $\mathcal{P}$ from 'finitary' meta-level systems $\mathcal{N}$ where:

- $\mathcal{P}$ can reason about crazy objects which cannot be constructed (and which would be rejected by intuitionists).
- $\mathcal{N}$ is grounded in a world of numbers and simple arithmetic operations, which no reasonable person could doubt. It is unquestionably correct.

The meta-level system $\mathcal{N}$ should be adequate for proving the consistency of the object-level system $\mathcal{P}$. In particular:

$$\mathcal{N} \vdash \text{“}\mathcal{P} \text{ is consistent”} \tag{1}$$

Unwinding Hilbert's formalism

The notion of 'finitary' in Hilbert's program is crucial and, indeed, a subject of debate. Hilbert asks us to distinguish object-level systems $\mathcal{P}$ from 'finitary' meta-level systems $\mathcal{N}$ where:

- $\mathcal{P}$ can reason about crazy objects which cannot be constructed (and which would be rejected by intuitionists).
- $\mathcal{N}$ is grounded in a world of numbers and simple arithmetic operations, which no reasonable person could doubt. It is unquestionably correct.

The meta-level system $\mathcal{N}$ should be adequate for proving the consistency of the object-level system $\mathcal{P}$. In particular:

$$\mathcal{N} \vdash \text{“}\mathcal{P} \text{ is consistent”} \tag{1}$$

Intuition: to trust $\mathcal{P}$, it is enough to trust $\mathcal{N}$.

Hilbert's formalist philsophy would identify, *e.g.*, the following two statements:

- Limits of monotone bounded sequences 'exist'.
- There is a 'proof' in an appropriate system $\mathcal{P}$ that every monotone bounded sequence has a limit. Moreover $\mathcal{P}$ satisfies (1), for an appropriate system $\mathcal{N}$.

Chasing dreams

Hilbert's program is a great idea! It is the gold standard of formalist philosophy, guaranteeing mathematical practice that is free from contradictions and paradoxes.

There is only one tiny catch...

Chasing dreams

Hilbert's program is a great idea! It is the gold standard of formalist philosophy, guaranteeing mathematical practice that is free from contradictions and paradoxes.

There is only one tiny catch...

... it doesn't work.

Chasing dreams

Hilbert's program is a great idea! It is the gold standard of formalist philosophy, guaranteeing mathematical practice that is free from contradictions and paradoxes.

There is only one tiny catch...

... it doesn't work.

ENTER K. GÖDEL:

Outline

Incompleteness

Gödel's First Incompletness Theorem, 1931

Any consistent formal system $\mathcal{P}$ over the language of arithmetic is incomplete: there are true statements that $\mathcal{P}$ cannot prove.

Gödel's First Incompletness Theorem, 1931

Any consistent formal system $\mathcal{P}$ over the language of arithmetic is incomplete: there are true statements that $\mathcal{P}$ cannot prove.

So Hilbert's program fails at the very first step! Namely, condition 1 fails.

How on earth did Gödel prove such a result?

Gödel's First Incompletness Theorem, 1931

Any consistent formal system $\mathcal{P}$ over the language of arithmetic is incomplete: there are true statements that $\mathcal{P}$ cannot prove.

So Hilbert's program fails at the very first step! Namely, condition 1 fails.

How on earth did Gödel prove such a result?

References for this part: [Raatikainen, 2018, Smullyan, 1992].

Setting the scene

Some conventions

- We consider $\mathcal{P}$ given by an *effective axiomatisation* over first-order logic.
- The underlying *language of arithmetic* contains the symbols $\mathsf{0}, \mathsf{s}, +, \times, \leq$.
- We assume that $\mathcal{P}$ proves at least some elementary properties of arithmetic. (This is only required later, for the 'second incompleteness' theorem)

($\mathsf{0}, +, \times, \leq$ have their usual interpretations over $\mathbb{N}$. $\mathsf{s}(x)$ is interpreted as $x + 1$.)

Setting the scene

Some conventions

- We consider $\mathcal{P}$ given by an *effective axiomatisation* over first-order logic.
- The underlying *language of arithmetic* contains the symbols $\mathsf{0}, \mathsf{s}, +, \times, \leq$.
- We assume that $\mathcal{P}$ proves at least some elementary properties of arithmetic. (This is only required later, for the 'second incompleteness' theorem)

($\mathsf{0}, +, \times, \leq$ have their usual interpretations over $\mathbb{N}$. $\mathsf{s}(x)$ is interpreted as $x + 1$.)

On consistency

What does it mean to be consistent? Three proposals (for now):

1. $\mathcal{P}$ does not prove φ and $\neg\varphi$, for any φ.
2. $\mathcal{P}$ does not prove $\bot$.
3. There is something that $\mathcal{P}$ does not prove.

Exercise: Prove that these are all, in fact, equivalent!

Self-reference in arithmetic

Remember Russell's paradox? It uses self-reference.

Self-reference in arithmetic

Remember Russell's paradox? It uses self-reference.

The language of arithmetic is quite expressive, and can formulate all sorts of self-reference:

Diagonal Lemma, informally

For any formula $\varphi(x)$, there is a sentence ψ such that $\psi \leftrightarrow \varphi(\ulcorner\psi\urcorner)$ (provably in $\mathcal{P}$).

Intuition: ψ is the sentence "I satisfy φ".

Self-reference in arithmetic

Remember Russell's paradox? It uses self-reference.

The language of arithmetic is quite expressive, and can formulate all sorts of self-reference:

Diagonal Lemma, informally

For any formula $\varphi(x)$, there is a sentence ψ such that $\psi \leftrightarrow \varphi(\ulcorner\psi\urcorner)$ (provably in $\mathcal{P}$).

Intuition: ψ is the sentence "I satisfy φ".

In particular, we may construct the sentence:

This sentence is not provable in $\mathcal{P}$.

Self-reference in arithmetic

Remember Russell's paradox? It uses self-reference.

The language of arithmetic is quite expressive, and can formulate all sorts of self-reference:

Diagonal Lemma, informally

For any formula $\varphi(x)$, there is a sentence ψ such that $\psi \leftrightarrow \varphi(\ulcorner\psi\urcorner)$ (provably in $\mathcal{P}$).

Intuition: ψ is the sentence "I satisfy φ".

In particular, we may construct the sentence:

This sentence is not provable in $\mathcal{P}$.

- If it were false, then it would be provable, which is bad.
- So it must be true, but then by definition it is not provable!

Self-reference in arithmetic

Remember Russell's paradox? It uses self-reference.

The language of arithmetic is quite expressive, and can formulate all sorts of self-reference:

Diagonal Lemma, informally

For any formula $\varphi(x)$, there is a sentence ψ such that $\psi \leftrightarrow \varphi(\ulcorner\psi\urcorner)$ (provably in $\mathcal{P}$).

Intuition: ψ is the sentence "I satisfy φ".

In particular, we may construct the sentence:

This sentence is not provable in $\mathcal{P}$.

- If it were false, then it would be provable, which is bad.
- So it must be true, but then by definition it is not provable!

Thus $\mathcal{P}$ does not prove every true sentence, yielding the first incompleteness theorem.

Digging deeper: construction of fixed points

Since there are only countably many formulae, we may enumerate them by a 'simple' injection $\ulcorner\cdot\urcorner : \mathsf{Formulae} \to \mathbb{N}$. $\ulcorner\varphi\urcorner$ is known as the *Gödel number* or *code* of φ.

Digging deeper: construction of fixed points

Since there are only countably many formulae, we may enumerate them by a 'simple' injection $\ulcorner\cdot\urcorner : \mathsf{Formulae} \to \mathbb{N}$. $\ulcorner\varphi\urcorner$ is known as the *Gödel number* or *code* of φ.

1. We can build a program $\mathsf{fix}(n)$ such that, for any formula $\chi(x)$,

$$\mathsf{fix}(\ulcorner\chi(x)\urcorner) = \ulcorner\chi(\ulcorner\chi(x)\urcorner)\urcorner$$

Intuition: $\mathsf{fix}(n)$ plugs the n^{th} formula into itself!

Digging deeper: construction of fixed points

Since there are only countably many formulae, we may enumerate them by a 'simple' injection $\ulcorner\cdot\urcorner : \mathsf{Formulae} \to \mathbb{N}$. $\ulcorner\varphi\urcorner$ is known as the *Gödel number* or *code* of φ.

1. We can build a program $\mathsf{fix}(n)$ such that, for any formula $\chi(x)$,

$$\mathsf{fix}(\ulcorner\chi(x)\urcorner) = \ulcorner\chi(\ulcorner\chi(x)\urcorner)\urcorner$$

   Intuition: $\mathsf{fix}(n)$ plugs the n^{th} formula into itself!

2. We may define the diagonal sentence $\psi := \varphi(\mathsf{fix}(\ulcorner\varphi(\mathsf{fix}(x))\urcorner))$.

Digging deeper: construction of fixed points

Since there are only countably many formulae, we may enumerate them by a 'simple' injection $\ulcorner\cdot\urcorner : \mathsf{Formulae} \to \mathbb{N}$. $\ulcorner\varphi\urcorner$ is known as the *Gödel number* or *code* of φ.

1. We can build a program $\mathsf{fix}(n)$ such that, for any formula $\chi(x)$,

$$\mathsf{fix}(\ulcorner\chi(x)\urcorner) = \ulcorner\chi(\ulcorner\chi(x)\urcorner)\urcorner$$

   Intuition: $\mathsf{fix}(n)$ plugs the n^{th} formula into itself!

2. We may define the diagonal sentence $\psi := \varphi(\mathsf{fix}(\ulcorner\varphi(\mathsf{fix}(x))\urcorner))$.

By inspecting the definitions we have:

$$\begin{array}{lrcll}
 & \psi & = & \varphi(\mathsf{fix}(\ulcorner\varphi(\mathsf{fix}(x))\urcorner)) & \text{by definition of } \psi \\
\text{so} & \ulcorner\psi\urcorner & = & \ulcorner\varphi(\mathsf{fix}(\ulcorner\varphi(\mathsf{fix}(x))\urcorner))\urcorner & \\
 & & = & \mathsf{fix}(\ulcorner\varphi(\mathsf{fix}(x))\urcorner) & \text{by definition of fix} \\
\text{so} & \psi & \iff & \varphi(\ulcorner\psi\urcorner) & \text{by definition of } \psi \text{ again}
\end{array}$$

Things can only get worse

Gödel's results went far deeper: not only are theories like arithmetic incomplete, they are unable to verify their own consistency.

Gödel's Second Incompletess Theorem, 1931

If $\mathcal{P}$ is consistent then the consistency of $\mathcal{P}$ cannot be proven within $\mathcal{P}$ itself.

Things can only get worse

Gödel's results went far deeper: not only are theories like arithmetic incomplete, they are unable to verify their own consistency.

Gödel's Second Incompletess Theorem, 1931

If $\mathcal{P}$ is consistent then the consistency of $\mathcal{P}$ cannot be proven within $\mathcal{P}$ itself.

This is even *worse* for Hilbert's program. Not only does condition 1 fail, even if it satisfies condition 2, even a very weak version of condition 3 fails too.

Things can only get worse

Gödel's results went far deeper: not only are theories like arithmetic incomplete, they are unable to verify their own consistency.

Gödel's Second Incompletess Theorem, 1931

If $\mathcal{P}$ is consistent then the consistency of $\mathcal{P}$ cannot be proven within $\mathcal{P}$ itself.

This is even *worse* for Hilbert's program. Not only does condition 1 fail, even if it satisfies condition 2, even a very weak version of condition 3 fails too.

Idea of argument: Formalise the proof of the first incompleteness theorem within $\mathcal{P}$ itself.

Provability and modal logic

Even weak systems $\mathcal{P}$ may formalise their own provability relation.

SOME MORE NOTATION

At the meta-level, we write $\mathcal{P} \vdash \varphi$ if φ is provable in the system $\mathcal{P}$. We may define the following arithmetic formula:

$$\mathsf{Prf}(x, y) \;:=\; \text{“}x \text{ is a description of a proof of the formula coded by } y\text{”}$$

Even weak systems $\mathcal{P}$ may formalise their own provability relation.

SOME MORE NOTATION

At the meta-level, we write $\mathcal{P} \vdash \varphi$ if φ is provable in the system $\mathcal{P}$. We may define the following arithmetic formula:

$$\mathsf{Prf}(x, y) \;:=\; \text{“}x \text{ is a description of a proof of the formula coded by } y\text{”}$$

Proposition (Hilbert-Bernays-Löb conditions)

Write $\Box\varphi$ *for* $\exists x.\mathsf{Prf}(x, \ulcorner\varphi\urcorner)$. *We have:*

(nec) *If* $\mathcal{P} \vdash \varphi$ *then* $\mathcal{P} \vdash \Box\varphi$.

(k) $\mathcal{P} \vdash \Box(\varphi \to \psi) \to (\Box\varphi \to \Box\psi)$.

(4) $\mathcal{P} \vdash \Box\varphi \to \Box\Box\varphi$.

“I am provable” is provable

Theorem (Löb, 1955)
If $\mathcal{P} \vdash \Box\varphi \to \varphi$ *then* $\mathcal{P} \vdash \varphi$.

"I am provable" is provable

Theorem (Löb, 1955)
If $\mathcal{P} \vdash \Box\varphi \to \varphi$ *then* $\mathcal{P} \vdash \varphi$.

Proof.
By the diagonal lemma, let ψ be a sentence such that:

$$\mathcal{P} \vdash \psi \leftrightarrow (\Box\psi \to \varphi) \tag{2}$$

We have, provably in $\mathcal{P}$:

$\psi \to (\Box\psi \to \varphi)$	by (2)($\to$)	$\Box\psi \to \Box\varphi$	by pure logic
$\Box(\psi \to (\Box\psi \to \varphi))$	by (nec)	$\Box\psi \to \varphi$	by assumption
$\Box\psi \to \Box(\Box\psi \to \varphi)$	by (k)	ψ	by (2)($\leftarrow$)
$\Box\psi \to (\Box\Box\psi \to \Box\varphi)$	by (k)	$\Box\psi$	by (nec)
$\Box\psi \to (\Box\psi \to \Box\varphi)$	by (4)	φ	by (*mp*). $\square$

Consistency: a matter of faith?

From Löb's theorem we immediately arrive at Gödel's second incompleteness result.

Corollary (Gödel's Second Incompleteness Theorem, again)

If $\mathcal{P}$ proves its own consistency, then $\mathcal{P}$ is, in fact, inconsistent.

Consistency: a matter of faith?

From Löb's theorem we immediately arrive at Gödel's second incompleteness result.

Corollary (Gödel's Second Incompleteness Theorem, again)

If $\mathcal{P}$ proves its own consistency, then $\mathcal{P}$ is, in fact, inconsistent.

(Let us write $\mathrm{Con}(\mathcal{P})$ for $\neg\Box\bot$.)

Proof.

We have:

$$\begin{array}{lll} \mathcal{P} \vdash \mathrm{Con}(\mathcal{P}) & \Longrightarrow & \mathcal{P} \vdash \Box\bot \to \bot \quad \text{by pure logic} \\ & \Longrightarrow & \mathcal{P} \vdash \bot \quad \text{by Löb's theorem.} \end{array}$$

□

Recap: what this means for Hilbert's program

For any consistent system $\mathcal{P}$:

- *$\mathcal{P}$ cannot be complete*. If it were then it would prove elementary arithmetic, which would render it vulnerable to Gödel's first incompleteness theorem.

Recap: what this means for Hilbert's program

For any consistent system $\mathcal{P}$:

- *$\mathcal{P}$ cannot be complete*. If it were then it would prove elementary arithmetic, which would render it vulnerable to Gödel's first incompleteness theorem.
- *$\mathcal{P}$ cannot prove its own consistency*, by the second incompleteness theorem. In particular, its consistency cannot be proved in any simpler subsystem.

Recap: what this means for Hilbert's program

For any consistent system $\mathcal{P}$:

- *$\mathcal{P}$ cannot be complete*. If it were then it would prove elementary arithmetic, which would render it vulnerable to Gödel's first incompleteness theorem.
- *$\mathcal{P}$ cannot prove its own consistency*, by the second incompleteness theorem. In particular, its consistency cannot be proved in any simpler subsystem.

Hilbert's program, in its absolute sense, is unachievable.

Reflection (a.k.a informal exercises)

BREAK

1. Show that the three formulations of 'consistency' are equivalent.
2. Can you identify, informally, what features of 'elementary arithmetic' we needed to formalise the second incompleteness theorem?
3. We showed how to construct single fixed points ('the diagonal lemma'). For formulae φ_0, φ_1, construct formulae ψ_0, ψ_1 such that:

$$\mathcal{P} \vdash \psi_0 \leftrightarrow \varphi(\ulcorner \psi_1 \urcorner)$$
$$\mathcal{P} \vdash \psi_1 \leftrightarrow \varphi(\ulcorner \psi_0 \urcorner)$$

   Can you generalise this further?
4. Construct an appropriate injection $\ulcorner \cdot \urcorner : \mathsf{Formulae} \to \mathbb{N}$. What properties of arithmetic do you need to express this function, and to 'decode' numbers?
5. Convince yourselves that proofs can be coded by numbers.

Outline

Reformulating Hilbert's program

Gödel's incompleteness theorems demonstrate that Hilbert's program could not be achieved, *but only in its most narrow sense*.

From a broader perspective, Hilbert's program was an extraordinary success, and continues in spirit in modern mathematics, particularly proof theory and theoretical computer science. Its legacy includes:

Reformulating Hilbert's program

Gödel's incompleteness theorems demonstrate that Hilbert's program could not be achieved, *but only in its most narrow sense*.

From a broader perspective, Hilbert's program was an extraordinary success, and continues in spirit in modern mathematics, particularly proof theory and theoretical computer science. Its legacy includes:

- The establishment of formalism: Proofs as syntactic objects which can be manipulated according to combinatorial rules.
- The connection between proofs and programs, and the development of powerful techniques which translate between the two.
- Introducing the idea that ordinary mathematical proofs can be studied as objects in their own right.

This course

This lecture course is about the success of Hilbert's program and the role it plays in research today.

We focus on **proof interpretations**: formal translations between strong logical theories $\mathcal{P}$ and 'finitary' systems $\mathcal{N}$

$$\mathcal{P} \mapsto \mathcal{N}$$

This course

This lecture course is about the success of Hilbert's program and the role it plays in research today.

We focus on **proof interpretations**: formal translations between strong logical theories $\mathcal{P}$ and 'finitary' systems $\mathcal{N}$

$$\mathcal{P} \mapsto \mathcal{N}$$

Our aims are to:

1. Introduce some well-known proof interpretations and explain how they work;
2. Emphasise the connection between proof and computation;
3. Explain how we can 'compute' fundamentally non-computable objects;
4. Highlight some exciting areas of research in which proof interpretations and ideas from Hilbert's program play a role today.

For the remainder of this lecture we give an broad outline of the main ideas.

Proof Interpretations

Recall the original aim of Hilbert's program: Reduce a mathematical theory $\mathcal{P}$ to a very simple finitary theory $\mathcal{N}$, which we imagine as a map

$$\mathcal{P} \mapsto \mathcal{N}$$

The aim is that we would then have

$$\mathrm{Con}(\mathcal{N}) \Rightarrow \mathrm{Con}(\mathcal{P}).$$

Proof Interpretations

Recall the original aim of Hilbert's program: Reduce a mathematical theory $\mathcal{P}$ to a very simple finitary theory $\mathcal{N}$, which we imagine as a map

$$\mathcal{P} \mapsto \mathcal{N}$$

The aim is that we would then have

$$\mathrm{Con}(\mathcal{N}) \Rightarrow \mathrm{Con}(\mathcal{P}).$$

In Hilbert's original formulation:

- $\mathcal{P}$ is a theory which contains a reasonable portion of 'ordinary mathematics', such as Peano arithmetic. We can use these theories to prove the existence of non-computable objects.
- $\mathcal{N}$ is a simple, finitistic theory. This was never fully specified by Hilbert, but the intuition was that it contains nothing beyond simple numbers and arithmetical operations, and is entirely computational.

Gödel's functional ('Dialectica') interpretation (1958)

One of the first responses to Hilbert's program was given by Gödel himself. His idea was to weaken the notion of 'finitary':

$$\underbrace{\mathrm{PA}}_{\mathcal{P}} \mapsto \underbrace{\text{System } \mathsf{T}}_{\mathcal{N}}$$

He achieved one of the first *relative* consistency proofs of ordinary mathematics, namely:

$$\mathrm{Con}(\mathsf{T}) \Rightarrow \mathrm{Con}(\mathrm{PA})$$

Gödel's functional ('Dialectica') interpretation (1958)

One of the first responses to Hilbert's program was given by Gödel himself. His idea was to weaken the notion of 'finitary':

$$\underbrace{\mathrm{PA}}_{\mathcal{P}} \mapsto \underbrace{\mathrm{System\ T}}_{\mathcal{N}}$$

He achieved one of the first *relative* consistency proofs of ordinary mathematics, namely:

$$\mathrm{Con}(\mathsf{T}) \Rightarrow \mathrm{Con}(\mathrm{PA})$$

In Gödel's consistency proof:

- $\mathcal{P}$ is the theory of Peano arithmetic;
- $\mathcal{N}$ is a new theory called System T. It contains simple operations on numbers, but also more complicated things, such as recursion over higher types, which are not strictly finitary in Hilbert's sense.
- System T is essentially a simple functional programming language akin to Haskell.

G. Kreisel and the 'unwinding of proofs'

To summarise:

- Gödel's functional interpretation is to interpret a complicated logical theory in a simpler calculus of programs (i.e. programming language).
- Originally, this was to obtain relative consistency proofs - to reduce the 'trustworthyness' of ordinary mathematics to that of a simple programming language.

G. Kreisel and the 'unwinding of proofs'

To summarise:

- Gödel's functional interpretation is to interpret a complicated logical theory in a simpler calculus of programs (i.e. programming language).
- Originally, this was to obtain relative consistency proofs - to reduce the 'trustworthyness' of ordinary mathematics to that of a simple programming language.

However, in the 1960s the Austrian logician G. Kreisel proposed using proof interpretations for a different purpose: to extract explicit computational information from existential statements, or more generally put:

> *"What more do we know if we have proved a theorem by restricted mean than if we merely know that it is true?"*

For example: If we have a concrete proof that an object x exists, can we analyse the proof to find some 'computational information' about x?

Computational information

What do we mean by 'computational information'?

- A proof of $P \vee Q \rightsquigarrow$ a boolean which tells us which of P or Q one is true.
- A proof that a set $X \subseteq \mathbb{N}$ is infinite $\rightsquigarrow$ for each $n \in \mathbb{N}$ a way to find some $m > n$ with $m \in X$.
- A proof that there is a real number x satisfying $R(x) \rightsquigarrow$ a method for computing approximations to x up to any desired accuracy.

Computational information

What do we mean by 'computational information'?

- A proof of $P \vee Q \rightsquigarrow$ a boolean which tells us which of P or Q one is true.
- A proof that a set $X \subseteq \mathbb{N}$ is infinite $\rightsquigarrow$ for each $n \in \mathbb{N}$ a way to find some $m > n$ with $m \in X$.
- A proof that there is a real number x satisfying $R(x) \rightsquigarrow$ a method for computing approximations to x up to any desired accuracy.

But what about those fundamentally non-constructive objects?

- A proof that a sequence converges $\rightsquigarrow$ A rate of convergence? Can we always find this?
- A proof that every vector space has a basis $\rightsquigarrow$???

This is a question central to the course.

Outline

System T (Lecture 2)

System T is a system of functionals.

It contains number $\mathbb{N}$, and allows us to construct recursive functions $\mathbb{N} \to \mathbb{N}$. So far, so good, everything looks 'finitary'.

System T also allows us to construct recursive functionals $(\mathbb{N} \to \mathbb{N}) \to \mathbb{N}$ i.e. functions which takes another function as an input. We can actually continue this

- $((\mathbb{N} \to \mathbb{N}) \to \mathbb{N}) \to \mathbb{N}$
- $(((\mathbb{N} \to \mathbb{N}) \to \mathbb{N}) \to \mathbb{N}) \to \mathbb{N}$
- $((((\mathbb{N} \to \mathbb{N}) \to \mathbb{N}) \to \mathbb{N}) \to \mathbb{N}) \to \mathbb{N}$ and so on ...

For this reason System T is not purely finitary (in the sense of Hilbert). Rather, it is an early example of a *programming language*!

System T (Lecture 2)

System T is a system of functionals.

It contains number $\mathbb{N}$, and allows us to construct recursive functions $\mathbb{N} \to \mathbb{N}$. So far, so good, everything looks 'finitary'.

System T also allows us to construct recursive functionals $(\mathbb{N} \to \mathbb{N}) \to \mathbb{N}$ i.e. functions which takes another function as an input. We can actually continue this

- $((\mathbb{N} \to \mathbb{N}) \to \mathbb{N}) \to \mathbb{N}$
- $(((\mathbb{N} \to \mathbb{N}) \to \mathbb{N}) \to \mathbb{N}) \to \mathbb{N}$
- $((((\mathbb{N} \to \mathbb{N}) \to \mathbb{N}) \to \mathbb{N}) \to \mathbb{N}) \to \mathbb{N}$ and so on ...

For this reason System T is not purely finitary (in the sense of Hilbert). Rather, it is an early example of a *programming language*!

Lecture 2 will introduce:

1. computable functions;
2. higher order functionals and System T;
3. the concept of recursion.

The functional interpretation of intuitionistic arithmetic (Lecture 3)

Gödel's intuitionistic functional interpretation maps theorems A in Heyting arithmetic (i.e. Peano arithmetic but without the law of excluded-middle $P \vee \neg P$) to formulas of the form $\exists x \forall y A_D(x, y)$ where $A_D(x, y)$ is a quantifier-free formula of System T. The main soundness theorem states:

$$\text{If HA proves } A \text{ then System T proves } \forall y A_D(t, y)$$

where t is a term (i.e. a program) that can be formally extracted from the proof of A.

In **Lecture 3** we will:

1. discuss the notion of program extraction in more detail;
2. briefly outline the theory of Heyting arithmetic;
3. study the interpretation $A \mapsto A_D(x, y)$;
4. give an overview of the soundness theorem;
5. carry out some worked examples.

Classical logic and the negative translation (Lecture 4)

There are some things which cannot be explicitly constructed e.g. a function f witnessing the Halting problem.

$$f(e,x) := \begin{cases} 1 & \text{if } \{e\} \text{ terminates on input } x \\ 0 & \text{otherwise} \end{cases}$$

One can prove that such a function *exists*, but no computable f can be constructed.

So how does the functional interpretation treat this kind of thing?

In **Lecture 4** we will:

1. give some examples of existential statements which are fundamentally non-computable;
2. describe the kind of 'indirect' information we can nevertheless extract from the underlying classical proofs;
3. make this formal via the negative translation of classical logic into intuitionistic logic;
4. discuss the special case of $\forall\exists$ statements.

Proof interpretations today (Lecture 5)

The application of proof interpretations in *mathematics* and *computer science* has taken off in the last two decades, and is an active and exciting area of research today. We will outline several topics at the cutting edge, and describe a range of open questions.

Lecture 5 will provide a high level snapshot of present day research in applied proof theory, including:

1. Proof mining - the extraction of quantitative information from proofs in mathematical analysis;
2. Extensions of proof interpretations to strong and weak theories;
3. The formalization of program extraction in proof assistants and the synthesis of verified programs;
4. Alternative techniques for extracting programs from proofs. Applications to computational complexity.

Outline

References I

Avigad, J. and Feferman, S. (1998).
Gödel's functional ("Dialectica") interpretation.
In Buss, S. R., editor, *Handbook of Proof Theory*, volume 137, pages 337–405. Elsevier.
http://www.andrew.cmu.edu/user/avigad/Papers/dialect.pdf.

Iemhoff, R. (2016).
Intuitionism in the philosophy of mathematics.
In Zalta, E. N., editor, *The Stanford Encyclopedia of Philosophy*. Metaphysics Research Lab, Stanford University, winter 2016 edition.

Kohlenbach, U. (2008).
Applied Proof Theory - Proof Interpretations and their Use in Mathematics.
Springer Monographs in Mathematics. Springer.

Raatikainen, P. (2018).
Gödel's incompleteness theorems.
In Zalta, E. N., editor, *The Stanford Encyclopedia of Philosophy*. Metaphysics Research Lab, Stanford University, summer 2018 edition.

Smullyan, R. M. (1992).
Gödel's Incompleteness Theorems.
Oxford Logic Guides. Oxford University Press.

References II

Weir, A. (2015).
Formalism in the philosophy of mathematics.
In Zalta, E. N., editor, *The Stanford Encyclopedia of Philosophy*. Metaphysics Research Lab, Stanford University, spring 2015 edition.